In [1]:
```
import numpy as np
from scipy import stats
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn import metrics
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
```

In [2]:
```
# data collection & data processing
# load dataset using pandas dataframe
sonar_data = pd.read_csv('sonar.csv', header=None)
```

In [3]:
```
# show data using pandas
sonar_data.head()
```

Out[3]:

	0	1	2	3	4	5	6	7	8	9	...	51	52	
0	0.0200	0.0371	0.0428	0.0207	0.0954	0.0986	0.1539	0.1601	0.3109	0.2111	...	0.0027	0.0065	0.
1	0.0453	0.0523	0.0843	0.0689	0.1183	0.2583	0.2156	0.3481	0.3337	0.2872	...	0.0084	0.0089	0.
2	0.0262	0.0582	0.1099	0.1083	0.0974	0.2280	0.2431	0.3771	0.5598	0.6194	...	0.0232	0.0166	0.
3	0.0100	0.0171	0.0623	0.0205	0.0205	0.0368	0.1098	0.1276	0.0598	0.1264	...	0.0121	0.0036	0.
4	0.0762	0.0666	0.0481	0.0394	0.0590	0.0649	0.1209	0.2467	0.3564	0.4459	...	0.0031	0.0054	0.

5 rows × 61 columns

In [4]:
```
# number of rows and columns
sonar_data.shape
```

Out[4]:
```
(208, 61)
```

In [5]:
```
# describe--> statistical measure of the data
sonar_data.describe()
```

Out[5]:

	0	1	2	3	4	5	6	7
count	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000
mean	0.029164	0.038437	0.043832	0.053892	0.075202	0.104570	0.121747	0.134799
std	0.022991	0.032960	0.038428	0.046528	0.055552	0.059105	0.061788	0.085152
min	0.001500	0.000600	0.001500	0.005800	0.006700	0.010200	0.003300	0.005500
25%	0.013350	0.016450	0.018950	0.024375	0.038050	0.067025	0.080900	0.080425
50%	0.022800	0.030800	0.034300	0.044050	0.062500	0.092150	0.106950	0.112100
75%	0.035550	0.047950	0.057950	0.064500	0.100275	0.134125	0.154000	0.169600
max	0.137100	0.233900	0.305900	0.426400	0.401000	0.382300	0.372900	0.459000

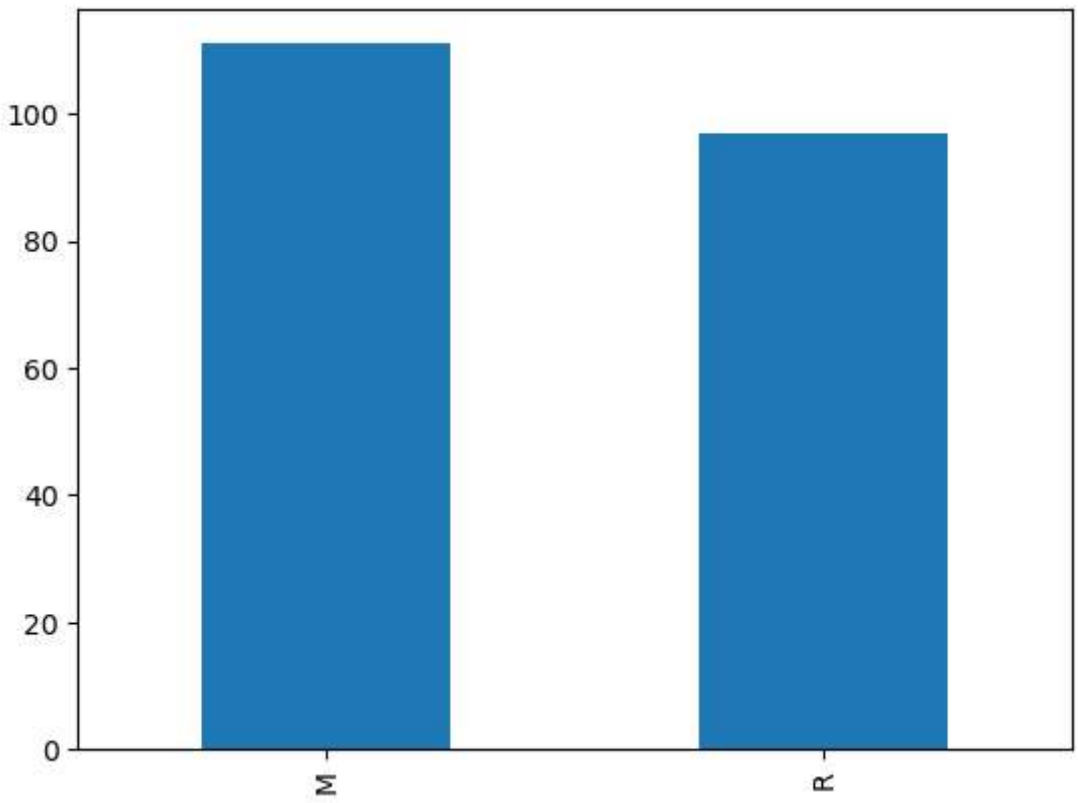
8 rows × 60 columns

In [6]:
```
sonar_data[60].value_counts()
```

Out[6]:
```
M    111
R     97
Name: 60, dtype: int64
```

In [7]:
```
sonar_data[60].value_counts().plot(kind='bar')
```

Out[7]:
```
<AxesSubplot:>
```



In [8]:
```
sonar_data.groupby(60).mean()
# m--> Mine
# R--> rock
```

Out[8]:

60	0	1	2	3	4	5	6	7	8	9
M	0.034989	0.045544	0.050720	0.064768	0.086715	0.111864	0.128359	0.149832	0.213492	0.251022
R	0.022498	0.030303	0.035951	0.041447	0.062028	0.096224	0.114180	0.117596	0.137392	0.159325

2 rows × 60 columns

In [9]:
```
# separating data and labels
# it is supervised learning to use labels data
x = sonar_data.drop(columns=60, axis=1)
y = sonar_data[60]
```

In [10]:
```
print(x)
print(y)
```

```
        0       1       2       3       4       5       6       7       8   \
0    0.0200  0.0371  0.0428  0.0207  0.0954  0.0986  0.1539  0.1601  0.3109
1    0.0453  0.0523  0.0843  0.0689  0.1183  0.2583  0.2156  0.3481  0.3337
2    0.0262  0.0582  0.1099  0.1083  0.0974  0.2280  0.2431  0.3771  0.5598
3    0.0100  0.0171  0.0623  0.0205  0.0205  0.0368  0.1098  0.1276  0.0598
4    0.0762  0.0666  0.0481  0.0394  0.0590  0.0649  0.1209  0.2467  0.3564
..      ...     ...     ...     ...     ...     ...     ...     ...     ...
203  0.0187  0.0346  0.0168  0.0177  0.0393  0.1630  0.2028  0.1694  0.2328
204  0.0323  0.0101  0.0298  0.0564  0.0760  0.0958  0.0990  0.1018  0.1030
205  0.0522  0.0437  0.0180  0.0292  0.0351  0.1171  0.1257  0.1178  0.1258
206  0.0303  0.0353  0.0490  0.0608  0.0167  0.1354  0.1465  0.1123  0.1945
207  0.0260  0.0363  0.0136  0.0272  0.0214  0.0338  0.0655  0.1400  0.1843

        9   ...      50      51      52      53      54      55      56  \
0    0.2111  ...  0.0232  0.0027  0.0065  0.0159  0.0072  0.0167  0.0180
1    0.2872  ...  0.0125  0.0084  0.0089  0.0048  0.0094  0.0191  0.0140
2    0.6194  ...  0.0033  0.0232  0.0166  0.0095  0.0180  0.0244  0.0316
3    0.1264  ...  0.0241  0.0121  0.0036  0.0150  0.0085  0.0073  0.0050
4    0.4459  ...  0.0156  0.0031  0.0054  0.0105  0.0110  0.0015  0.0072
..      ...  ...     ...     ...     ...     ...     ...     ...     ...
203  0.2684  ...  0.0203  0.0116  0.0098  0.0199  0.0033  0.0101  0.0065
204  0.2154  ...  0.0051  0.0061  0.0093  0.0135  0.0063  0.0063  0.0034
205  0.2529  ...  0.0155  0.0160  0.0029  0.0051  0.0062  0.0089  0.0140
206  0.2354  ...  0.0042  0.0086  0.0046  0.0126  0.0036  0.0035  0.0034
207  0.2354  ...  0.0181  0.0146  0.0129  0.0047  0.0039  0.0061  0.0040

        57      58      59
0    0.0084  0.0090  0.0032
1    0.0049  0.0052  0.0044
2    0.0164  0.0095  0.0078
3    0.0044  0.0040  0.0117
4    0.0048  0.0107  0.0094
..      ...     ...     ...
203  0.0115  0.0193  0.0157
204  0.0032  0.0062  0.0067
205  0.0138  0.0077  0.0031
206  0.0079  0.0036  0.0048
207  0.0036  0.0061  0.0115

[208 rows x 60 columns]
0      R
1      R
2      R
3      R
4      R
      ..
203    M
204    M
205    M
206    M
207    M
Name: 60, Length: 208, dtype: object
```

In [11]:
```
# training and test data
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, stratify =
```

In [12]:
```
print(x.shape, x_train.shape, x_test.shape)
```

```
(208, 60) (187, 60) (21, 60)
```

In [13]:
```
print(x_train)
print(y_train)
```

```
        0       1       2       3       4       5       6       7       8   \
115  0.0414  0.0436  0.0447  0.0844  0.0419  0.1215  0.2002  0.1516  0.0818
38   0.0123  0.0022  0.0196  0.0206  0.0180  0.0492  0.0033  0.0398  0.0791
56   0.0152  0.0102  0.0113  0.0263  0.0097  0.0391  0.0857  0.0915  0.0949
123  0.0270  0.0163  0.0341  0.0247  0.0822  0.1256  0.1323  0.1584  0.2017
18   0.0270  0.0092  0.0145  0.0278  0.0412  0.0757  0.1026  0.1138  0.0794
..      ...     ...     ...     ...     ...     ...     ...     ...     ...
140  0.0412  0.1135  0.0518  0.0232  0.0646  0.1124  0.1787  0.2407  0.2682
5    0.0286  0.0453  0.0277  0.0174  0.0384  0.0990  0.1201  0.1833  0.2105
154  0.0117  0.0069  0.0279  0.0583  0.0915  0.1267  0.1577  0.1927  0.2361
131  0.1150  0.1163  0.0866  0.0358  0.0232  0.1267  0.2417  0.2661  0.4346
203  0.0187  0.0346  0.0168  0.0177  0.0393  0.1630  0.2028  0.1694  0.2328

        9   ...      50      51      52      53      54      55      56  \
115  0.1975  ...  0.0222  0.0045  0.0136  0.0113  0.0053  0.0165  0.0141
38   0.0475  ...  0.0149  0.0125  0.0134  0.0026  0.0038  0.0018  0.0113
56   0.1504  ...  0.0048  0.0049  0.0041  0.0036  0.0013  0.0046  0.0037
123  0.2122  ...  0.0197  0.0189  0.0204  0.0085  0.0043  0.0092  0.0138
18   0.1520  ...  0.0045  0.0084  0.0010  0.0018  0.0068  0.0039  0.0120
..      ...  ...     ...     ...     ...     ...     ...     ...     ...
140  0.2058  ...  0.0798  0.0376  0.0143  0.0272  0.0127  0.0166  0.0095
5    0.3039  ...  0.0104  0.0045  0.0014  0.0038  0.0013  0.0089  0.0057
154  0.2169  ...  0.0039  0.0053  0.0029  0.0020  0.0013  0.0029  0.0020
131  0.5378  ...  0.0228  0.0099  0.0065  0.0085  0.0166  0.0110  0.0190
203  0.2684  ...  0.0203  0.0116  0.0098  0.0199  0.0033  0.0101  0.0065

        57      58      59
115  0.0077  0.0246  0.0198
38   0.0058  0.0047  0.0071
56   0.0011  0.0034  0.0033
123  0.0094  0.0105  0.0093
18   0.0132  0.0070  0.0088
..      ...     ...     ...
140  0.0225  0.0098  0.0085
5    0.0027  0.0051  0.0062
154  0.0062  0.0026  0.0052
131  0.0141  0.0068  0.0086
203  0.0115  0.0193  0.0157

[187 rows x 60 columns]
115    M
38     R
56     R
123    M
18     R
      ..
140    M
5      R
154    M
131    M
203    M
Name: 60, Length: 187, dtype: object
```

In [ ]:
```
```

In [14]:
```
#model traning ==> logistic regression
model = LogisticRegression()
```

In [15]:
```
# Train the model on the training data
model.fit(x_train, y_train)
```

Out[15]:
```
LogisticRegression()
```

In [16]:
```
#accuracy on training data
x_train_prediction = model.predict(x_train)
x_train_accuracy = accuracy_score(x_train_prediction, y_train)
```

In [17]:
```
print('accuracy on training data: ', x_train_accuracy)
```

```
accuracy on training data:  0.8342245989304813
```

In [18]:
```
#accuracy on test data
x_test_prediction = model.predict(x_test)
x_test_accuracy = accuracy_score(x_test_prediction, y_test)
```

In [19]:
```
print('accuracy on test data: ', x_test_accuracy)
```

```
accuracy on test data:  0.7619047619047619
```

In [20]:
```
input_data = [0.0115,0.0150,0.0136,0.0076,0.0211,0.1058,0.1023,0.0440,0.0931,0.0734,0.

# Convert input data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# Reshape data
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

# Make a prediction using your model
prediction = model.predict(input_data_reshaped)

# Print the prediction
print(prediction)
if (prediction[0]== "R"):
  print('The object is a ROCK')
else:
    print('The Object is MINE')
```

```
['R']
The object is a ROCK
```

In [21]:
```
input_data = [0.0180,0.0444,0.0476,0.0698,0.1615,0.0887,0.0596,0.1071,0.3175,0.2918,0.

# Convert input data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# Reshape data
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

# Make a prediction using your model
prediction = model.predict(input_data_reshaped)

# Print the prediction
print(prediction)
```

```
if (prediction[0]== "M"):
  print('The object is a MINE')
else:
    print('The Object is ROCK')
```

```
['M']
The object is a MINE
```

In [22]:
```
# Define a KNN classifier with 5 neighbors
classifier= KNeighborsClassifier(n_neighbors=3, metric='minkowski', p=2)
classifier.fit(x_train, y_train)

#Predicting the result
x_pred_train= classifier.predict(x_train)
print("KNN model accuracy of training data is (in %):", metrics.accuracy_score(y_trair
```

```
KNN model accuracy of training data is (in %): 89.3048128342246
C:\Users\Jawad Ahmad\anaconda3\lib\site-packages\sklearn\neighbors\_classification.p
y:228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the
default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` will become False, the
`axis` over which the statistic is taken will be eliminated, and the value None will
no longer be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

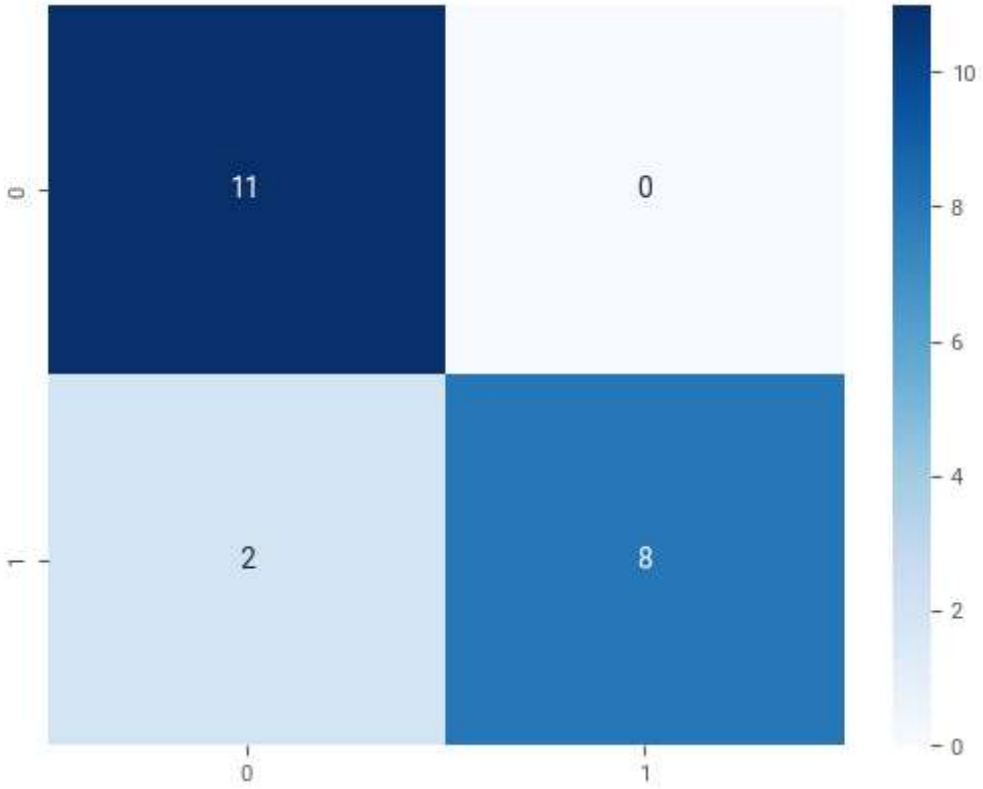
In [23]:
```
x_pred_test= classifier.predict(x_test)
print("KNN model accuracy of test data is (in %):", metrics.accuracy_score(y_test, x_p
```

```
KNN model accuracy of test data is (in %): 90.47619047619048
C:\Users\Jawad Ahmad\anaconda3\lib\site-packages\sklearn\neighbors\_classification.p
y:228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the
default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` will become False, the
`axis` over which the statistic is taken will be eliminated, and the value None will
no longer be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [37]:
```
from sklearn.metrics import confusion_matrix
import seaborn as sns

# Assuming you have trained a KNN model and made predictions on your test data
knn_predictions = classifier.predict(x_test)

# Creating a confusion matrix
cm = confusion_matrix(y_test, knn_predictions)

# Plotting the confusion matrix as a heatmap using seaborn
sns.heatmap(cm, annot=True, cmap="Blues", fmt="d")
```

```
C:\Users\Jawad Ahmad\anaconda3\lib\site-packages\sklearn\neighbors\_classification.p
y:228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the
default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` will become False, the
`axis` over which the statistic is taken will be eliminated, and the value None will
no longer be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

Out[37]:
```
<AxesSubplot:>
```



In [34]:
```
import pandas as pd
!pip install pandas-profiling
from pandas_profiling import ProfileReport

# Load the sonar dataset
df = pd.read_csv('sonar.csv')

# Create a profile report
profile = ProfileReport(df, title='Sonar Dataset Profile Report')

# Save the report as an HTML file
profile.to_file(output_file='sonar_report.html')
```

```
Requirement already satisfied: pandas-profiling in c:\users\jawad ahmad\anaconda3\lib
\site-packages (3.6.6)
Requirement already satisfied: ydata-profiling in c:\users\jawad ahmad\anaconda3\lib
\site-packages (from pandas-profiling) (4.1.2)
Requirement already satisfied: typeguard<2.14,>=2.13.2 in c:\users\jawad ahmad\anacon
da3\lib\site-packages (from ydata-profiling->pandas-profiling) (2.13.3)
Requirement already satisfied: pandas!=1.4.0,<1.6,>1.1 in c:\users\jawad ahmad\anacon
da3\lib\site-packages (from ydata-profiling->pandas-profiling) (1.4.4)
Requirement already satisfied: phik<0.13,>=0.11.1 in c:\users\jawad ahmad\anaconda3\l
ib\site-packages (from ydata-profiling->pandas-profiling) (0.12.3)
Requirement already satisfied: matplotlib<3.7,>=3.2 in c:\users\jawad ahmad\anaconda3
\lib\site-packages (from ydata-profiling->pandas-profiling) (3.5.2)
Requirement already satisfied: pydantic<1.11,>=1.8.1 in c:\users\jawad ahmad\anaconda
3\lib\site-packages (from ydata-profiling->pandas-profiling) (1.10.7)
Requirement already satisfied: jinja2<3.2,>=2.11.1 in c:\users\jawad ahmad\anaconda3
\lib\site-packages (from ydata-profiling->pandas-profiling) (2.11.3)
Requirement already satisfied: requests<2.29,>=2.24.0 in c:\users\jawad ahmad\anacond
a3\lib\site-packages (from ydata-profiling->pandas-profiling) (2.28.1)
Requirement already satisfied: statsmodels<0.14,>=0.13.2 in c:\users\jawad ahmad\anac
onda3\lib\site-packages (from ydata-profiling->pandas-profiling) (0.13.2)
Requirement already satisfied: visions[type_image_path]==0.7.5 in c:\users\jawad ahma
d\anaconda3\lib\site-packages (from ydata-profiling->pandas-profiling) (0.7.5)
Requirement already satisfied: scipy<1.10,>=1.4.1 in c:\users\jawad ahmad\anaconda3\l
ib\site-packages (from ydata-profiling->pandas-profiling) (1.9.1)
Requirement already satisfied: numpy<1.24,>=1.16.0 in c:\users\jawad ahmad\anaconda3
\lib\site-packages (from ydata-profiling->pandas-profiling) (1.21.5)
Requirement already satisfied: htmlmin==0.1.12 in c:\users\jawad ahmad\anaconda3\lib
\site-packages (from ydata-profiling->pandas-profiling) (0.1.12)
Requirement already satisfied: seaborn<0.13,>=0.10.1 in c:\users\jawad ahmad\anaconda
3\lib\site-packages (from ydata-profiling->pandas-profiling) (0.11.2)
Requirement already satisfied: multimethod<1.10,>=1.4 in c:\users\jawad ahmad\anacond
a3\lib\site-packages (from ydata-profiling->pandas-profiling) (1.9.1)
Requirement already satisfied: imagehash==4.3.1 in c:\users\jawad ahmad\anaconda3\lib
\site-packages (from ydata-profiling->pandas-profiling) (4.3.1)
Requirement already satisfied: tqdm<4.65,>=4.48.2 in c:\users\jawad ahmad\anaconda3\l
ib\site-packages (from ydata-profiling->pandas-profiling) (4.64.1)
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in c:\users\jawad ahmad\anaconda3\l
ib\site-packages (from ydata-profiling->pandas-profiling) (6.0)
Requirement already satisfied: PyWavelets in c:\users\jawad ahmad\anaconda3\lib\site-
packages (from imagehash==4.3.1->ydata-profiling->pandas-profiling) (1.3.0)
Requirement already satisfied: pillow in c:\users\jawad ahmad\anaconda3\lib\site-pack
ages (from imagehash==4.3.1->ydata-profiling->pandas-profiling) (9.2.0)
Requirement already satisfied: networkx>=2.4 in c:\users\jawad ahmad\anaconda3\lib\si
te-packages (from visions[type_image_path]==0.7.5->ydata-profiling->pandas-profiling)
(2.8.4)
Requirement already satisfied: tangled-up-in-unicode>=0.0.4 in c:\users\jawad ahmad\a
naconda3\lib\site-packages (from visions[type_image_path]==0.7.5->ydata-profiling->pa
ndas-profiling) (0.2.0)
Requirement already satisfied: attrs>=19.3.0 in c:\users\jawad ahmad\anaconda3\lib\si
te-packages (from visions[type_image_path]==0.7.5->ydata-profiling->pandas-profiling)
(21.4.0)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\jawad ahmad\anaconda3\lib
\site-packages (from jinja2<3.2,>=2.11.1->ydata-profiling->pandas-profiling) (2.0.1)
Requirement already satisfied: cycler>=0.10 in c:\users\jawad ahmad\anaconda3\lib\sit
e-packages (from matplotlib<3.7,>=3.2->ydata-profiling->pandas-profiling) (0.11.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\jawad ahmad\anaconda3\lib
\site-packages (from matplotlib<3.7,>=3.2->ydata-profiling->pandas-profiling) (3.0.9)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\jawad ahmad\anaconda3\li
b\site-packages (from matplotlib<3.7,>=3.2->ydata-profiling->pandas-profiling) (1.4.
2)
```

```
Requirement already satisfied: python-dateutil>=2.7 in c:\users\jawad ahmad\anaconda3
\lib\site-packages (from matplotlib<3.7,>=3.2->ydata-profiling->pandas-profiling) (2.
8.2)
Requirement already satisfied: packaging>=20.0 in c:\users\jawad ahmad\anaconda3\lib
\site-packages (from matplotlib<3.7,>=3.2->ydata-profiling->pandas-profiling) (21.3)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\jawad ahmad\anaconda3\li
b\site-packages (from matplotlib<3.7,>=3.2->ydata-profiling->pandas-profiling) (4.25.
0)
Requirement already satisfied: pytz>=2020.1 in c:\users\jawad ahmad\anaconda3\lib\sit
e-packages (from pandas!=1.4.0,<1.6,>1.1->ydata-profiling->pandas-profiling) (2022.1)
Requirement already satisfied: joblib>=0.14.1 in c:\users\jawad ahmad\anaconda3\lib\s
ite-packages (from phik<0.13,>=0.11.1->ydata-profiling->pandas-profiling) (1.1.0)
Requirement already satisfied: typing-extensions>=4.2.0 in c:\users\jawad ahmad\anaco
nda3\lib\site-packages (from pydantic<1.11,>=1.8.1->ydata-profiling->pandas-profilin
g) (4.3.0)
Requirement already satisfied: idna<4,>=2.5 in c:\users\jawad ahmad\anaconda3\lib\sit
e-packages (from requests<2.29,>=2.24.0->ydata-profiling->pandas-profiling) (3.3)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\jawad ahmad\anaco
nda3\lib\site-packages (from requests<2.29,>=2.24.0->ydata-profiling->pandas-profilin
g) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\jawad ahmad\anaconda3\l
ib\site-packages (from requests<2.29,>=2.24.0->ydata-profiling->pandas-profiling) (20
22.9.14)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\jawad ahmad\anaconda
3\lib\site-packages (from requests<2.29,>=2.24.0->ydata-profiling->pandas-profiling)
(1.26.11)
Requirement already satisfied: patsy>=0.5.2 in c:\users\jawad ahmad\anaconda3\lib\sit
e-packages (from statsmodels<0.14,>=0.13.2->ydata-profiling->pandas-profiling) (0.5.
2)
Requirement already satisfied: colorama in c:\users\jawad ahmad\anaconda3\lib\site-pa
ckages (from tqdm<4.65,>=4.48.2->ydata-profiling->pandas-profiling) (0.4.5)
Requirement already satisfied: six in c:\users\jawad ahmad\anaconda3\lib\site-package
s (from patsy>=0.5.2->statsmodels<0.14,>=0.13.2->ydata-profiling->pandas-profiling)
(1.16.0)
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
Export report to file:   0%|          | 0/1 [00:00<?, ?it/s]
```

In [35]:

```
import seaborn as sns
import pandas as pd
sonar_data = pd.read_csv('sonar.csv')
!pip install sweetviz
import sweetviz as sd
d=sd.analyze(sonar_data)
print(d.show_html())
```

```
Requirement already satisfied: sweetviz in c:\users\jawad ahmad\anaconda3\lib\site-pa
ckages (2.1.4)
Requirement already satisfied: numpy>=1.16.0 in c:\users\jawad ahmad\anaconda3\lib\si
te-packages (from sweetviz) (1.21.5)
Requirement already satisfied: matplotlib>=3.1.3 in c:\users\jawad ahmad\anaconda3\li
b\site-packages (from sweetviz) (3.5.2)
Requirement already satisfied: jinja2>=2.11.1 in c:\users\jawad ahmad\anaconda3\lib\s
ite-packages (from sweetviz) (2.11.3)
Requirement already satisfied: importlib-resources>=1.2.0 in c:\users\jawad ahmad\ana
conda3\lib\site-packages (from sweetviz) (5.12.0)
Requirement already satisfied: tqdm>=4.43.0 in c:\users\jawad ahmad\anaconda3\lib\sit
e-packages (from sweetviz) (4.64.1)
Requirement already satisfied: scipy>=1.3.2 in c:\users\jawad ahmad\anaconda3\lib\sit
e-packages (from sweetviz) (1.9.1)
Requirement already satisfied: pandas!=1.0.0,!=1.0.1,!=1.0.2,>=0.25.3 in c:\users\jaw
ad ahmad\anaconda3\lib\site-packages (from sweetviz) (1.4.4)
Requirement already satisfied: zipp>=3.1.0 in c:\users\jawad ahmad\anaconda3\lib\site
-packages (from importlib-resources>=1.2.0->sweetviz) (3.8.0)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\jawad ahmad\anaconda3\lib
\site-packages (from jinja2>=2.11.1->sweetviz) (2.0.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\jawad ahmad\anaconda3\li
b\site-packages (from matplotlib>=3.1.3->sweetviz) (4.25.0)
Requirement already satisfied: cycler>=0.10 in c:\users\jawad ahmad\anaconda3\lib\sit
e-packages (from matplotlib>=3.1.3->sweetviz) (0.11.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\jawad ahmad\anaconda3\lib
\site-packages (from matplotlib>=3.1.3->sweetviz) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\jawad ahmad\anaconda3
\lib\site-packages (from matplotlib>=3.1.3->sweetviz) (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\jawad ahmad\anaconda3\li
b\site-packages (from matplotlib>=3.1.3->sweetviz) (1.4.2)
Requirement already satisfied: pillow>=6.2.0 in c:\users\jawad ahmad\anaconda3\lib\si
te-packages (from matplotlib>=3.1.3->sweetviz) (9.2.0)
Requirement already satisfied: packaging>=20.0 in c:\users\jawad ahmad\anaconda3\lib
\site-packages (from matplotlib>=3.1.3->sweetviz) (21.3)
Requirement already satisfied: pytz>=2020.1 in c:\users\jawad ahmad\anaconda3\lib\sit
e-packages (from pandas!=1.0.0,!=1.0.1,!=1.0.2,>=0.25.3->sweetviz) (2022.1)
Requirement already satisfied: colorama in c:\users\jawad ahmad\anaconda3\lib\site-pa
ckages (from tqdm>=4.43.0->sweetviz) (0.4.5)
Requirement already satisfied: six>=1.5 in c:\users\jawad ahmad\anaconda3\lib\site-pa
ckages (from python-dateutil>=2.7->matplotlib>=3.1.3->sweetviz) (1.16.0)
                                                |
| [  0%]   00:00 ->…
Report SWEETVIZ_REPORT.html was generated! NOTEBOOK/COLAB USERS: the web browser MAY
not pop up, regardless, the report IS saved in your notebook/colab files.
None
```

In [ ]: